DBMS LAB CHIT SOLUTIONS

(1) Chit 4 –

```
use india
db.createCollection("city")

db.city.insert([
...    {
...       "city": "Pune",
...       "type": "Urban",
...       "state": "MH",
...       "population": 5600000
...    },
...    {
...       "city": "Mumbai",
...       "type": "Urban",
...       "state": "MH",
...       "population": 78100000
...    },
...    {
...       "city": "Nashik",
...       "type": "Urban",
...       "state": "MH",
...       "population": 54761000
...    },
...    {
...       "city": "Ahmedabad",
...       "type": "Urban",
...       "state": "GJ",
...       "population": 23145670
...    },
...    {
...       "city": "Surat",
...       "type": "Urban",
...       "state": "GJ",
...       "population": 45179410
...    },
...    {
...       "city": "aurangabad",
...       "type": "Rural",
...       "state": "MH",
...       "population": 12000000
...    },
...    {
...       "city": "Thane",
...       "type": "Rural",
```

```
...         "state": "MH",
...         "population": 19000000
...     },
...     {
...         "city": "Deoli",
...         "type": "Rural",
...         "state": "MH",
...         "population": 16000
...     },
...     {
...         "city": "Bhopal",
...         "type": "Urban",
...         "state": "MP",
...         "population": 18000000
...     },
...     {
...         "city": "Ashoknagar",
...         "type": "Rural",
...         "state": "MP",
...         "population": 845071
...     }
... ])
```

QUERIES

- **-using mapreduce, find statewise population**

  ```
  var map = function() {emit(this.state,this.population)}
  var reduce = function(key, values) {return Array.sum(values)}
  db.city.mapReduce(map, reduce, {out : 'statewise'})
  db.statewise.find().pretty()
  ```

- **-using mapreduce, find citywise population**

  ```
  var map = function() {emit(this.city, this.population)}

  var reduce = function(key, values) {return Array.sum(values)}

  db.city.mapReduce(map, reduce, {out : 'citywise'})

  db.citywise.find().pretty()
  ```

- **-using mapreduce, find typewise population.**

  ```
  var map = function() {emit(this.type, this.population)}
  var reduce = function(key, values) {return Array.sum(values)}
  db.city.mapReduce(map, reduce, {out : 'typewise'})
  db.typewise.find().pretty()
  ```

- **-using mapreduce , find citywise count.**

```
var map = function() {emit(this.city, this._id)}
var reduce = function(key, values) {return values.length}
db.city.mapReduce(map, reduce, {out : 'citywise_count'})
db.citywise_count.find().pretty()
```

- **using mapreduce , find statewise count.**

```
var map = function() {emit(this.state, this._id)}
var reduce = function(key, values) {return values.length}
db.city.mapReduce(map, reduce, {out : 'statewise_count'})
db.statewise_count.find().pretty()
```

(2) Chit 8 –

```
use college
db.createCollection("students")

db.students.insert([
...    {
...       "rno": "1",
...       "name": "Omkar",
...       "subject": "Geometry",
...       "class": "A",
...       "fees": 50000,
...       "marks": 90
...    },
...    {
...       "rno": "2",
...       "name": "Rahul",
...       "subject": "Science",
...       "class": "A",
...       "fees": 60000,
...       "marks": 80
...    },
...    {
...       "rno": "3",
...       "name": "Yash",
...       "subject": "Algebra",
...       "class": "A",
...       "fees": 30000,
...       "marks": 85
...    },
...    {
...       "rno": "4",
...       "name": "Ved",
```

```
...         "subject": "History",
...         "class": "A",
...         "fees": 40000,
...         "marks": 93
...     },
...     {
...         "rno": "5",
...         "name": "Shree",
...         "subject": "Geography",
...         "class": "A",
...         "fees": 35000,
...         "marks": 79
...     },
...     {
...         "rno": "1",
...         "name": "Yajur",
...         "subject": "Geometry",
...         "class": "B",
...         "fees": 45000,
...         "marks": 90
...     },
...     {
...         "rno": "2",
...         "name": "Ganraj",
...         "subject": "Science",
...         "class": "B",
...         "fees": 55000,
...         "marks": 87
...     },
...     {
...         "rno": "3",
...         "name": "Pratham",
...         "subject": "Algebra",
...         "class": "B",
...         "fees": 63000,
...         "marks": 97
...     },
...     {
...         "rno": "4",
...         "name": "Shreyash",
...         "subject": "History",
...         "class": "B",
...         "fees": 30000,
...         "marks": 88
...     },
...     {
...         "rno": "5",
...         "name": "Raj",
```

```
...         "subject": "Geography",
...         "class": "B",
...         "fees": 42000,
...         "marks": 85
...     }
... ])
```

QUERIES

- **Class wise total number of students**

  ```
  var map = function() {emit(this.class, this._id)}
  var reduce = function(key, values) {return values.length}
  db.students.mapReduce(map, reduce, {out: 'classwise_total'})
  db.classwise_total.find().pretty()
  ```

- **Class wise total fees**

  ```
  var map = function() {emit(this.class, this.fees)}
  var reduce = function(key, values) {return Array.sum(values)}
  db.students.mapReduce(map, reduce, {out: 'classwise_fees'})
  db.classwise_fees.find().pretty()
  ```

- **Subject wise total marks**

  ```
  var map = function() {emit(this.subject, this.marks)}
  var reduce = function(key, values) {return Array.sum(values)}
  db.students.mapReduce(map, reduce, {out: 'subjectwise_marks'})
  db.subjectwise_marks.find().pretty()
  ```

- **Subject wise total students**

  ```
  var map = function() {emit(this.subject, this._id)}
  var reduce = function(key, values) {return values.length}
  db.students.mapReduce(map, reduce, {out: 'subjectwise_total'})
  db.subjectwise_total.find().pretty()
  ```

(3) Chit 9 –

```
use orders
db.createCollection("orderinfo")

db.orderinfo.insert([
...     {
...         "cust_id": 123,
...         "cust_name": "abc",
...         "status": "A",
...         "price": 250
...     },
```

```
...   {
...     "cust_id": 123,
...     "cust_name": "abc",
...     "status": "A",
...     "price": 300
...   },
...   {
...     "cust_id": 123,
...     "cust_name": "abc",
...     "status": "A",
...     "price": 450
...   },
...   {
...     "cust_id": 456,
...     "cust_name": "xyz",
...     "status": "B",
...     "price": 600
...   },
...   {
...     "cust_id": 456,
...     "cust_name": "xyz",
...     "status": "B",
...     "price": 100
...   },
...   {
...     "cust_id": 456,
...     "cust_name": "xyz",
...     "status": "A",
...     "price": 900
...   },
...   {
...     "cust_id": 789,
...     "cust_name": "def",
...     "status": "A",
...     "price": 300
...   },
...   {
...     "cust_id": 789,
...     "cust_name": "def",
...     "status": "A",
...     "price": 150
...   },
...   {
...     "cust_id": 789,
...     "cust_name": "def",
...     "status": "B",
...     "price": 850
...   },
...   {
...     "cust_id": 789,
...     "cust_name": "def",
...     "status": "B",
```

```
...       "price": 1000
...    }
... ])
```

QUERIES

- **Find the total price for each customer and display in the order of total price.**

```
db.orderinfo.aggregate(
... [
...   { $group:
...           {_id: "$cust_id",
...            total_price:
...                     {$sum: "$price"}
...           }
...   },
...   { $sort:
...           {total_price: 1}
...   }
... ])
```

- **Find the distinct customer names**

```
db.orderinfo.distinct("cust_name")
```

- **Display the “price” of customers whose status is 'A'**

```
db.orderinfo.find(
... { status: 'A' },
... { cust_name: 1, price: 1, _id: 0 }
... ).pretty()
```

- **Delete the customers whose status is 'A'**

```
db.orderinfo.remove({status: 'A'})
```

(4) Chit 14 –

```
use college_db
db.createCollection("teachers")
db.createCollection("departments")
db.createCollection("students")

db.departments.insert([
...    {
...       "dno": 1,
...       "dname": "Computer"
...    },
...    {
...       "dno": 2,
```

```
...         "dname": "It"
...     },
...     {
...         "dno": 3,
...         "dname": "Entc"
...     },
...     {
...         "dno": 4,
...         "dname": "Mechanical"
...     },
...     {
...         "dno": 1,
...         "dname": "Civil"
...     }
... ])

db.teachers.insert([
    {
        "tname": "Rahul",
        "dno": {
            "$ref": "departments",
            "$id": ObjectId("6379e1b2a8c2f73b2d061dd5")
        },
        "experience": 4,
        "salary": 40000,
        "date_of_joining": "2018-03-12"
    },
    {
        "tname": "Sonali",
        "dno": {
            "$ref": "departments",
            "$id": ObjectId("6379e1b2a8c2f73b2d061dd5")
        },
        "experience": 10,
        "salary": 90000,
        "date_of_joining": "2016-02-14"
    },
    {
        "tname": "Vijay",
        "dno": {
            "$ref": "departments",
            "$id": ObjectId("6379e1b2a8c2f73b2d061dd7")
        },
        "experience": 5,
        "salary": 50000,
        "date_of_joining": "2019-04-16"
    },
    {
        "tname": "Archana",
        "dno": {
            "$ref": "departments",
            "$id": ObjectId("6379e1b2a8c2f73b2d061dd7")
```

```
    },
    "experience": 12,
    "salary": 100000,
    "date_of_joining": "2013-07-19"
  },
  {
    "tname": "Tushar",
    "dno": {
      "$ref": "departments",
      "$id": ObjectId("6379e1b2a8c2f73b2d061dd8")
    },
    "experience": 8,
    "salary": 95000,
    "date_of_joining": "2017-05-17"
  },
  {
    "tname": "Manisha",
    "dno": {
      "$ref": "departments",
      "$id": ObjectId("6379e1b2a8c2f73b2d061dd8")
    },
    "experience": 10,
    "salary": 80000,
    "date_of_joining": "2016-01-26"
  },
  {
    "tname": "Priya",
    "dno": {
      "$ref": "departments",
      "$id": ObjectId("6379e1b2a8c2f73b2d061dd9")
    },
    "experience": 7,
    "salary": 75000,
    "date_of_joining": "2015-04-26"
  },
  {
    "tname": "Sachin",
    "dno": {
      "$ref": "departments",
      "$id": ObjectId("6379e1b2a8c2f73b2d061dd6")
    },
    "experience": 5,
    "salary": 50000,
    "date_of_joining": "2020-05-15"
  }
])


db.students.insert([
...   {
...     "sname": "Varun",
...     "rollno": 4,
```

```
...     "class": "A"
...   },
...   {
...     "sname": "Aditya",
...     "rollno": 12,
...     "class": "B"
...   },
...   {
...     "sname": "Om",
...     "rollno": 14,
...     "class": "A"
...   },
...   {
...     "sname": "Rajat",
...     "rollno": 19,
...     "class": "B"
...   },
...   {
...     "sname": "Prathamesh",
...     "rollno": 41,
...     "class": "A"
...   },
...   {
...     "sname": "Aditya",
...     "rollno": 42,
...     "class": "B"
...   },
...   {
...     "sname": "Abhishek",
...     "rollno": 45,
...     "class": "A"
...   },
...   {
...     "sname": "Atharva",
...     "rollno": 58,
...     "class": "B"
...   }
... ])
```

QUERIES

- **Find the information about all teachers of Dno=2 and having salary greater than or equal to 10,000/-**

```
db.teachers.aggregate([
  {
    $lookup: {
      from: "departments",
      localField: "dno.$id",
      foreignField: "_id",
      as: "dno"
    }
```

```
    },
    {
        $match: {
            "dno.dno":2,
            salary: {$gte: 10000}
            }
    },
    {
        $project: {
            tname: 1,
            "dno.dno": 1,
            "dno.dname": 1,
            experience: 1,
            salary: 1,
            date_of_joining: 1,
            _id: 0
            }
    }
]).pretty()
```

- **Find the student information having Roll_no=2 or Sname='xyz'**

  db.students.find({$or: [{roll_no: 2}, {sname: 'xyz'}]}).pretty()

- **Update student name whose Roll_No=5**

  db.students.update({roll_no: 5}, {$set: {sname: 'Shree'}})

- **Delete all student whose Class is 'FE'**

  db.students.remove({class: 'FE'})

- **Find information of Teachers whose Experience is more than 10 years**

  db.teachers.find({experience: {$gt: 10}}).pretty()

- **Apply index on Students Collection**

  db.students.createIndex({roll_no: 1})

(5) Chit 15 –

- **Create database Institute.**
  use Institute

- **Create collection Students.**
  db.createCollection('students')

- **Insert 10 documents with above mentioned structure.**

```
db.students.insert([
{
```

```
        "roll_no": 1,
        "stud_name": "A",
        "age": 10,
        "branch": "Comp",
        "address": {
                "city": "Pune",
                "state": "MH"
        },
         "hobbies": [
                "playing",
                "reading"
        ]
},
{
        "roll_no": 2,
        "stud_name": "B",
        "age": 11,
        "branch": "Comp",
        "address": {
                "city": "Mumbai",
                "state": "MH"
        },
        "hobbies": [
                "reading"
        ]
},
{
        "roll_no": 3,
        "stud_name": "C",
        "age": 12,
        "branch": "Civil",
        "address": {
                "city": "Nashik",
                "state": "MH"
        },
        "hobbies": [
                "reading",
                "singing"
        ]
},
{
        "roll_no": 4,
        "stud_name": "D",
        "age": 13,
        "branch": "Mech",
        "address": {
                "city": "Pune",
                "state": "MH"
        },
        "hobbies": [
                "singing",
                "dancing"
```

```
        ]
},
{
        "roll_no": 5,
        "stud_name": "E",
        "age": 14,
        "branch": "Comp",
        "address": {
                "city": "Mumbai",
                "state": "MH"
},
        "hobbies": [
                "singing"
        ]
},
{
        "roll_no": 6,
        "stud_name": "F",
        "age": 15,
        "branch": "E&TC",
        "address": {
                "city": "Pune",
                "state": "MH"
        },
        "hobbies": [
                "dancing"
        ]
},
{
        "roll_no": 7,
        "stud_name": "G",
        "age": 16,
        "branch": "IT",
        "address": {
                "city": "Pune",
                "state": "MH"
        },
        "hobbies": [
                "playing",
                "reading",
                "singing"
        ]
},
{
        "roll_no": 8,
        "stud_name": "H",
        "age": 17,
        "branch": "Comp",
        "address": {
                "city": "Pune",
                "state": "MH"
        },
```

```
        "hobbies": [
                "playing",
                "reading",
                "drawing"
        ]
    },
    {
        "roll_no": 9,
        "stud_name": "G",
        "age": 18,
        "branch": "Civil",
        "address": {
                "city": "Pune",
                "state": "MH"
        },
        "hobbies": [
                 "drumming"
        ]
    },
    {
        "roll_no": 10,
        "stud_name": "H",
        "age": 19,
        "branch": "Mech",
        "address": {
                "city": "Nashik",
                "state": "MH"
        },
        "hobbies": [
                "calling"
        ]
    }
])
```

- **Display all students' information.**
  db.students.find().pretty()

- **Display Student information whose age is greater than 15.**
  db.students.find({age: {$gt: 15}}).pretty()

- **Display Student information sorted on name field**
  db.students.find().sort({stud_name: 1}).pretty()

- **Update student branch Computer of RollNo 3.**
  db.students.update({roll_no: 3}, {$set: {branch: 'Comp'}})

- **Remove document with RollNo 1**
  db.students.remove({roll_no: 1})

- **Display Student information whose name starts with A**
  db.students.find({stud_name: \^A\}).pretty()

- **Display the total numbers of documents available in collection.**

```
db.students.aggregate(
[
        {
                $group: {
                                _id: '$roll_no',
                                total_documents : {$sum: 1}
                        }
        }
]).pretty()
```

- **Display only first 5 documents.**

```
db.students.find().limit(5).pretty()
```

- **Display all documents instead of first 3.**

```
db.students.find().skip(3).pretty()
```

- **Display the name of Students who live in Pune City.**

```
db.students.find({'address.city': 'Pune'}, {stud_name: 1, _id: 0}).pretty()
```

- **Display the list of different cities from where students are coming.**

```
db.students.distinct('address.city')
```

- **Display the list of different cities with number of students from belonging to that city.**

```
db.students.aggregate(
[
        {
                $group: {
                                _id: '$address.city',
                                total_students: {$sum: 1}
                        }
        }
]).pretty()
```

- **Display only Name of all students.**

```
db.students.find({}, {_id: 0, stud_name: 1}).pretty()
```

- **Display the hobbies of each student.**

```
db.students.find({}, {_id: 0, stud_name: 1, hobbies: 1}).pretty()
```

- **Drop Collection**

```
db.students.drop()
```

(6) Chit 16 –

```
use department
db.createCollection("teachers")
```

```
db.teachers.insert([
...  {
...    "name": "Rahul",
...    "department": "Computer",
...    "experience": 4,
...    "salary": 40000
...  },
...  {
...    "name": "Sonali",
...    "department": "Computer",
...    "experience": 10,
...    "salary": 90000
...  },
...  {
...    "name": "Vijay",
...    "department": "Entc",
...    "experience": 5,
...    "salary": 50000
...  },
...  {
...    "name": "Archana",
...    "department": "Entc",
...    "experience": 12,
...    "salary": 100000
...  },
...  {
...    "name": "Tushar",
...    "department": "It",
...    "experience": 8,
...    "salary": 95000
...  },
...  {
...    "name": "Manisha",
...    "department": "It",
...    "experience": 10,
...    "salary": 80000
...  },
...  {
...    "name": "Priya",
...    "department": "Civil",
...    "experience": 7,
...    "salary": 75000
...  },
...  {
...    "name": "Sachin",
...    "department": "Mechanical",
...    "experience": 5,
...    "salary": 50000
...  }
... ])
```

QUERIES

- **Display the department wise average salary.**

```
db.teachers.aggregate(
... [
...     {
...         $group: {
...                 _id: "$department",
...                 avg_salary: {$avg: "$salary"}
...             }
...     }
... ]).pretty()
```

- **Display the no. Of employees working in each department.**

```
db.teachers.aggregate(
... [
...     {
...         $group: {
...                 _id: "$department",
...                 no_of_employees: {$sum: 1}
...             }
...     }
... ]).pretty()
```

- **Display the department wise minimum salary.**

```
db.teachers.aggregate(
... [
...     {
...         $group: {
...                 _id: "$department",
...                 min_salary: {$min: "$salary"}
...             }
...     }
... ]).pretty()
```

- **Apply index and drop index**

```
db.teachers.createIndex({name: 1})

db.teachers.dropIndex({name: 1})
```

(7) Chit 20 –

use my_db

- **Create Collection**
  db.createCollection("students")

- **Insert some documents**

```
db.students.insert([
... {
...   "rollno": 1,
...   "name": "Navin",
...   "subject": "DMSA",
...   "marks": 78
... },
... {
...   "rollno": 2,
...   "name": "Anusha",
...   "subject": "OSD",
...   "marks": 75
... },
... {
...   "rollno": 3,
...   "name": "Ravi",
...   "subject": "TOC",
...   "marks": 69
... },
... {
...   "rollno": 4,
...   "name": "Veena",
...   "subject": "TOC",
...   "marks": 70
... },
... {
...   "rollno": 5,
...   "name": "Pravini",
...   "subject": "OSD",
...   "marks": 80
... },
... {
...   "rollno": 6,
...   "name": "Reena",
...   "subject": "DMSA",
...   "marks": 50
... },
... {
...   "rollno": 7,
...   "name": "Geeta",
...   "subject": "CN",
...   "marks": 90
... },
... {
```

```
...     "rollno": 8,
...     "name": "Akash",
...     "subject": "CN",
...     "marks": 85
...   }
... ])
```

- **Create Single Index.**

  db.students.createIndex({rollno: 1})

- **Create Compound Index.**

  db.students.createIndex({subject: 1, name: 1})

- **Create Unique on Collection**

  db.students.createIndex({name: 1}, {unique: true})

- **Show Index Information**

  db.students.getIndexes()

- **Remove Index**

  db.students.dropIndex({rollno: 1})

- **Write aggregate function to find Max marks of Each Subject.**

```
db.students.aggregate(
... [
...     {
...         $group: {
...                 _id: "$subject",
...                 max_marks: {$max: "$marks"}
...               }
...     }
... ]).pretty()
```

- **Write aggregate function to find Min marks of Each Subject.**

```
db.students.aggregate(
... [
...     {
...         $group: {
...                 _id: "$subject",
...                 min_marks: {$min: "$marks"}
...               }
...     }
... ]).pretty()
```

- **Write aggregate function to find Sum of marks of Each Subject.**

```
db.students.aggregate(
... [
...   {
...     $group: {
...             _id: "$subject",
...             sum_marks: {$sum: "$marks"}
...           }
...   }
... ]).pretty()
```

- **Write aggregate function to find Avg of marks of Each Subject.**

```
db.students.aggregate(
... [
...   {
...     $group: {
...             _id: "$subject",
...             avg_marks: {$avg: "$marks"}
...           }
...   }
... ]).pretty()
```

- **Write aggregate function to find first record of Each Subject.**

```
db.students.aggregate(
... [
...   {
...     $group: {
...             _id: "$subject",
...             first_record: {$first: "$subject"}
...           }
...   }
... ]).pretty()
```

- **Write aggregate function to find last record of Each Subject.**

```
db.students.aggregate(
... [
...   {
...     $group: {
...             _id: "$subject",
...             last_record: {$last: "$subject"}
...           }
...   }
... ]).pretty()
```

- **Write aggregate function to find count number of records of each subject**

```
db.students.aggregate(
... [
...   {
...     $group: {
...             _id: "$subject",
...             no_of_subjects: {$sum: 1}
...           }
...   }
... ]).pretty()
```

(8) Chit 24 –

```
use my_db
db.createCollection("movies")

db.movies.insert([
...  {
...   "name": "KGF",
...   "type": "action",
...   "budget": 1000000,
...   "producer": {
...    "name": "Raju",
...    "address": "Pune"
...   }
...  },
...  {
...   "name": "Bahubali",
...   "type": "mythical",
...   "budget": 2000000,
...   "producer": {
...    "name": "S.S. Rajamouli",
...    "address": "Pune"
...   }
...  },
...  {
...   "name": "Avatar",
...   "type": "Sci-fi",
...   "budget": 1000000000,
...   "producer": {
...    "name": "James",
...    "address": "Mumbai"
...   }
...  },
...  {
...   "name": "RRR",
...   "type": "adventure",
...   "budget": 3000000,
...   "producer": {
...    "name": "Danayya",
```

```
...      "address": "Mumbai"
...    }
...  },
...  {
...    "name": "Kantara",
...    "type": "thriller",
...    "budget": 5000000,
...    "producer": {
...      "name": "Vijay",
...      "address": "Pune"
...    }
...  }
... ])
```

QUERIES

- **Find the name of the movie having budget greater than 1,00,000.**

  ```
  db.movies.find({budget: {$gt: 100000}}).pretty()
  ```

- **Find the name of producer who lives in Pune**

  ```
  db.movies.find({"producer.address": "Pune"}, {"producer.name": 1, _id: 0}).pretty()
  ```

- **Update the type of movie “action” to “horror”**

  ```
  db.movies.update({type: 'action'}, {$set: {type: 'horror'}})
  ```

- **Find all the documents produced by name “producer1” with their address**

  ```
  db.movies.find({"producer.name": "producer1"}).pretty()
  ```

- **write any query using aggregate function – No of movies of each producer**

  ```
  db.movies.aggregate(
  ... [
  ...    {
  ...       $group: {
  ...                 _id: "$producer.name",
  ...                 no_of_movies: {$sum: 1}
  ...               }
  ...    }
  ... ]).pretty()
  ```